Imprecise Weighted Extensions of Random Forests for Classification and Regression

Lev V. Utkin[1], Maxim S. Kovalev[1] and Frank P.A. Coolen[2]
[1]Peter the Great St.Petersburg Polytechnic University (SPbPU), Russia
[2]Durham University, United Kingdom

Abstract

One of the main problems of using the random forests (RF) in classification and regression tasks is a lack of sufficient data which fall into certain leaves of trees in order to estimate the tree predicted values. To cope with this problem, robust imprecise classification and regression RF models, called the imprecise RF, are proposed. They are based on the following ideas. First, imprecision of the tree estimates is taken into account by means of imprecise statistical inference models and confidence interval models. Secondly, we introduce weights assigned to trees or to groups of trees, which are computed in order to correct the RF estimates under condition of imprecise tree predicted values. In fact, the weights can be regarded as a robust meta-learner controlling the imprecision of estimates. Special modifications of loss functions to compute optimal weights for the classification and regression tasks are proposed in order to simplify maximin optimization problems. As a result, simple linear and quadratic optimization problems are obtained, whose solution does not meet any difficulties. Various numerical examples with real datasets illustrate the proposed robust models and show outperforming results when datasets are rather small or noisy.

Keywords: classification, regression, random forest, decision tree, imprecise Dirichlet model, linear-vacuous mixture, confidence intervals, quadratic programming

1 Introduction

The ensemble methodology can be regarded as one of the efficient machine learning approaches for classification and regression. Therefore, a huge amount of ensemble-based methods for solving machine learning problems, including classification and regression, have been developed in recent years [16, 33]. These methods are based on constructing the so-called weak or base classifiers from training data, and on aggregating their predictions when classifying unknown samples in order to obtain a strong classifier that outperforms every single one of them. One of the best known and most effective ensemble-based models

is the Random Forest (RF) [10], which uses a large number of randomly built individual decision trees in order to combine their predictions. RFs reduce the possible correlation between decision trees by selecting different subsamples of the feature space. Depending on the solved machine learning problem, outputs of decision trees may vary. In particular, the probability distributions of classes are computed in classification problems. They are estimated by counting the percentage of different classes of examples at the leaf node where the concerned example falls into, and then are combined in order to get the class probability distribution of the corresponding RF. In regression problems, outputs of decision trees are usually represented in the form of predicted values corresponding to feature vectors which are averaged across all trees to get the RF prediction. The common combination procedure is the standard averaging of all tree class probability distributions in classification or predicted values in regression.

For the classification problem, the class probability distributions computed for every decision tree and for the RF is an informative and useful measure for classification because it provides a flexible decision. Therefore, the class probability distributions are widely used in many applications and the RF modifications. However, one of the main problems for using this measure is that the class probability distributions are assumed to be precise. This is quite restrictive, in particular if there is only a small amount of training data. It should be noted that it is difficult to expect the precise probabilities at some leaf nodes even when there are a lot of data, but only a few examples fall into these leaf nodes. Moreover, testing data may be noisy and differ from training data. This is also a reason of incorrect classification results and the reduced accuracy. This makes it interesting to consider a generalization by using imprecise probabilities or imprecise statistical inference models [39].

The same can be said about the regression problem. In this case, the leaf nodes are characterized by the predicted values which can be viewed as average values of outputs of training examples. If there is only a small number of training examples, then one cannot have much trust in resulting average values and may wish to consider corresponding confidence intervals instead.

One of the ways to get more accurate prediction for the RF is to introduce weights of trees or subsets of trees and to use the weighted average for computing the RF predictions. The weights can be viewed as additional training parameters and assigned to every tree according to the tree prediction accuracy (see, for example, [38, 37]). An important obstacle of using the weighted average as well as the simple average of the tree predictions is that they are assumed to be precise. However, the precision is not expected, especially in cases with only a small amount of training data. In order to overcome this obstacle, robust imprecise classification and regression RF models, called the imprecise RF (IRF), are proposed in this paper. One of the main ideas underlying the proposed model is to use the weights assigned to decision trees as a function of imprecise decision tree predictions. It is very important to point out that the weights are not used in order to improve the RF performance. They are used to get a maximin or robust decision about the predicted values of the RF under condition that the decision tree predictions are imprecise or interval-valued. In fact, the

weights can be regarded as a robust meta-learner controlling the imprecision of estimates. It is studied in the paper how the use of imprecise distributions in classification, or confidence intervals of predicted values in regression, may impact on the choice of the corresponding weights of trees.

In contrast to some works dealing with imprecise information in RFs, for example, [2, 3], where new special splitting rules were considered under condition of the noise data, a quite different approach is proposed, which does not change decision trees in order to take into account the available imprecision of estimations. Weights of the trees are trained in a way that allows us to control the imprecision and to implement the robust strategy of decision making. In fact, the weights can be regarded as some kind of regularization which improves imprecision accounting. At that, the proposed approach can also be applied to modified decision trees with the special splitting rules. The approach just improves the accounting of imprecision used in the splitting rule.

It is important to emphasize that trees are not changed for taking into account the available imprecision of estimates, but the weights are determined in the weighting aggregation procedures to account for the imprecision. This is the main idea underlying the proposed robust models. So, robust modifications of the classification and regression RFs taking into account imprecision of the decision tree estimates are proposed. The important ideas underlying the methods presented in this paper are as follows:

1. Aggregation procedures for computing the RF class probability distributions (classification task) and for computing the RF predicted values (regression task) are modified by introducing the weights of trees.

2. The imprecision of the tree estimates is defined by means of imprecise statistical inference models and confidence interval models, for example, by using the IDM for the classification task and confidence intervals for the regression task.

3. Special modifications of loss functions for the classification and regression tasks are proposed in order to simplify minimax and maximin optimization problems for computing optimal weights.

4. The obtained optimization problems are linear or quadratic with linear constraints.

The paper is organized as follows. The first part (Sections 2 and 3) is devoted to studying classification models based on the RF and the IRF. In particular, Section 2 describes a general approach to assign weights to decision trees in the RF. Implementation of ideas underlying the IRF for classification is given in Section 3. The second part (Section 4) considers the proposed approach for constructing the regression IRF. Various numerical examples with well-known public real data are presented in Section 5. These illustrate good performance of the IRF for classification and regression. Concluding remarks are provided in Section 6.

2 Related work

Ensemble-based models. A comprehensive description of ensemble-based models is presented in Zhou's book [45]. Analysis and comparison of many ensemble-based methods can also be found in several review papers and books. For example, [6] provided the R implementation of many ensemble-based models. Comprehensive surveys of these models can be found in [18, 20, 29, 30, 35, 42]. Ensemble models with convolutional neural networks are studied in [24].

A detailed analysis of RFs is presented by Rokach [34]. A review of ensemble methods in bioinformatics is presented by Yang et al. [44]. Due to many advantages of the RF as an ensemble-based model, a large amount of its modifications have been developed, for example, [5, 7, 14, 15, 23, 32, 41].

Imprecise probabilities in classification and regression. One of the first ideas of applying imprecise probability theory to classification decision trees was presented in [4], where probabilities of classes at decision tree leaves are estimated by using an imprecise model, and the so-called Credal Decision Tree model is proposed. Following this work, several papers devoted to applications of imprecise probabilities to decision trees and RFs were presented [2, 3, 25, 28], where the authors developed new splitting criteria taking into account imprecision of training data and noisy data. In particular, the authors consider the application of Walley's imprecise Dirichlet model (IDM) [40]. The main advantage of the IDM in its application to the classification problems is that it produces a convex set of probability distributions, which has nice properties and depends on a number of observations. Another interesting RF called the fuzzy RF is proposed in [9]. As an alternative to the use of the IDM, nonparametric predictive inference has also been used successfully for imprecise probabilistic inference with decision trees [1]. Imprecise probabilities have also been used in classification problems in [13, 26, 27]. The main focus of interest in this paper is not the decision trees or RFs, but the weighted averaging procedure which is used to combine the class probability distributions.

Weighted RFs. Rules for assigning the weights may be different. For example, the weights may be assigned to minimize the difference between class labels of training examples and values of class vectors which are formed by using the decision tree outputs. Optimal weights assigned to sets of the class probability distributions which are produced by every decision tree for all training examples, but not the weights of decision trees themselves, are considered in [37]. It should be pointed out that the idea of weighting trees in RFs is not new. Most weighting RF methods use weights of classes to deal with imbalanced data sets [11]. At the same time, there are a lot of papers devoted to more complex weight assignments to every tree [19, 21, 36]. These methods use some measures of the classification quality in order to assign the weights. In particular, a weighted voting classification ensemble method, called WAVE, is presented in [19], which uses two weight vectors: a weight vector of classifiers and a weight vector of examples. The example weight vector assigns higher weights to observations that are hard to classify. Methods to increase the classification accuracy of RFs through weighting trees according to their classification ability are

proposed in [21, 36]. However, the above papers consider the classification performance of every tree separately, and they do not take into account the lack of sufficient training data.

3 Weighted RFs for Classification

3.1 Weighted Averages

In order to consider weighted averaging in RFs, the standard classification problem is formally stated. Given N training data (examples, instances, patterns) $S = \{(\mathbf{x}_1, y_1), ..., (\mathbf{x}_N, y_N)\}$, in which $\mathbf{x}_i$ may belong to an arbitrary set $\mathcal{X}$ and represents a feature vector involving m features and $y_i \in \mathcal{Y} = \{1, ..., C\}$ represents the class of the associated examples, the task of classification is to construct an accurate classifier $c : \mathcal{X} \to \mathcal{Y}$ which can predict the unknown class label y of a new observation $\mathbf{x}$, using available training data, as accurately as possible such that it minimizes the classification risk defined as the following expectation of the loss function $l : \mathbb{R} \times \mathcal{Y} \to \mathbb{R}_+$: $\mathbb{E}_{(X,Y)\sim p(\mathbf{x},y)} [l(c(X), Y)]$, where $p(\mathbf{x}, y)$ is a joint density.

RFs can be regarded as a powerful nonparametric statistical method for both regression and classification problems. Suppose that the RF consists of T trained trees. One of the important peculiarities of decision trees is a probability distribution of classes at each leaf node. This probability distribution is used for computing probabilities of classes for the RF and for making decisions about a class label of a testing example. Formally, each leaf node $l \in CL = \{1, ..., L\}$ stores votes for the class labels denoted as $\mathbf{n}_l = (n_{l,1}, ..., n_{l,C})$. Here $n_{l,c}$ is the number of feature vectors from the class c which fall into the l-th leaf node. This is equivalent to storing a categorical probability distribution over classes $c \in \{1, ..., C\}$ in a vector $p_l = (p_{l,1}, ..., p_{l,C}) \in [0, 1]^C$. If vector $\mathbf{x}$ falls into the l-th leaf node in a tree, then the prediction of the decision tree for feature vector $\mathbf{x}$ to be of class c is given by $p_{l,c} = n_{l,c}/n_l$. Here n_l is the number of all feature vectors which fall into the l-th leaf node.

One of the ways to improve RFs is to assign weights to decision trees. The weights are used in order to compute the weighted average of the class probability distributions across all trees. They are regarded as training parameters. Their values should minimize the difference between class labels of training examples and values of the RF class distributions. This objective stems from the following reasoning. If an example $\mathbf{x}_i$ has the label $y_i = c$, then it is expected that the c-th element of the forest class probability distribution should be close to 1, and other elements of the vector are close to 0. Of course, this condition may be violated. However, this violation could be reduced by controlling the weights for computing the forest class distributions. So, weights of trees could be found in order to minimize the mean difference between class vectors of all examples and the corresponding forest class probability distributions, i.e., the weights can be trained by solving the corresponding optimization problem.

The c-th element $v_{i,c}$ of the class probability distribution produced by the RF for $\mathbf{x}_i$ is

determined as

$$v_{i,c} = \frac{1}{T} \sum\nolimits_{t=1}^{T} p_{i,c}^{(t)}, \tag{1}$$

where $p_{i,c}^{(t)}$ is the probability of class c for $\mathbf{x}_i$ produced by the t-th tree from the RF.

Denote the obtained RF class probability distribution as $\mathbf{v}_i = (v_{i,1}, ..., v_{i,C})$. It is proposed to change the method for computing $v_{i,c}$, namely, the averaging operator (1) is replaced with the weighted sum with weights $\mathbf{w} = (w_1, ..., w_T)$ of the form:

$$v_{i,c} = \sum\nolimits_{t=1}^{T} p_{i,c}^{(t)} w_t, \tag{2}$$

where w_t is a weight of the t-th tree.

The weights do not depend on the class c. They are identical for all classes, but different for trees. The weights also do not depend on $\mathbf{x}_i$. They are restricted by the following obvious condition:

$$\sum\nolimits_{t=1}^{T} w_t = \mathbf{w} \cdot \mathbf{1}^{\mathrm{T}} = 1, \ w_t \geq 0, \ t = 1, ..., T, \tag{3}$$

where $\mathbf{1}$ is a vector having T unit elements.

3.2 Training the Weighted RF Classifier

The RF output for $\mathbf{x}_i$, which is denoted by $\mathbf{o}_i$, would be ideal when the class probability distribution $\mathbf{v}_i$ is such that it contains a single unit element and $C - 1$ zero elements. If the feature vector $\mathbf{x}_i$ belongs to the class y_i, then the class vector is

$$\mathbf{o}_i = (0, ..., 0, 1_{y_i}, 0, ..., 0).$$

Of course, the ideal RF for all training elements cannot be constructed, but the trees can be supplemented by a weighted averaging procedure that could try to approximate the class probability distribution of the RF to $\mathbf{o}_i$ for every example $\mathbf{x}_i$. Weights $\mathbf{w}$ are found such that vectors $\mathbf{v}_i$ will be as close as possible to vectors $\mathbf{o}_i$ whose unit element has index y_i coinciding with the class label of $\mathbf{x}_i$. This can be done by minimizing the loss function which is defined by the distance $d(\mathbf{v}_i, \mathbf{o}_i)$ between $\mathbf{v}_i$ and $\mathbf{o}_i$, i.e.,

$$\min_{\mathbf{w}} \left(\sum\nolimits_{i=1}^{N} d(\mathbf{v}_i, \mathbf{o}_i) + \lambda R(\mathbf{w}) \right), \tag{4}$$

where $R(\mathbf{w})$ is a regularization term which aims to avoid overfitting; λ is a hyper-parameter which controls the strength of the regularization.

The regularization term is taken of the form $R(\mathbf{w}) = \|\mathbf{w}\|^2$ in order to get a quadratic optimization problem.

A simple way to solve the optimization problem (4) is to use the Euclidean distance between two vectors. As a result, the loss function for every RF is rewritten as follows:

$$\min_{\mathbf{w}} \left(\sum_{i=1}^{N} \sum_{c=1}^{C} \left(\mathbf{p}_{i,c}\mathbf{w} - I_c(y_i) \right)^2 + \lambda \|\mathbf{w}\|^2 \right), \tag{5}$$

subject to (3).

Here $I_c(y)$ is the indicator function taking the value 1 if $y = c$, otherwise it is 0; $\mathbf{p}_{i,c} = \left(p_{i,c}^{(1)}, ..., p_{i,c}^{(T)}\right)$. This is a standard convex optimization problem with linear constraints whose solution does not meet any difficulties.

It follows from (3) that weights are restricted by the unit simplex. Generally, the set of weights can be restricted by some convex subset $\mathcal{W}(u)$ of the unit simplex in order to improve the regularization. Here u is another regularization parameter which defines the size of the subset $\mathcal{W}$, for example, the hyperparameter of the imprecise Dirichlet model [40] if this model is used for producing the subset $\mathcal{W}(u)$. One of the aims of regularization is to restrict a set of solutions to the optimization problem and to smooth it in order to avoid outliers. The introduction of the restriction $\mathcal{W}(u)$ for the set of weights plays the same role as the regularization. In summary, it will be assumed that constraints to problem (5) are of the form $\mathbf{w} \in \mathcal{W}(u)$.

4 Probabilities of Classes for Trees and Imprecise Probability Models

It is obvious that estimates of class probabilities cannot be considered to be precise when based on a small number of training data. Even if there are a lot of training examples, it does not guarantee that many examples fall into a certain leaf node, i.e., n_l is large for all $l \in CL$. This implies that interval-valued or imprecise probabilities $p_{l,c}$ should be taken in place of the precise ones.

Suppose that training example $\mathbf{x}_i$ produces a class probability distribution $\mathbf{P}(i,t) = (p_1(i,t), ..., p_C(i,t))$ at a leaf node of the t-th decision tree, and this distribution is not precisely known, but it is known that it belongs to a set $\mathcal{P}_{i.t}(s)$. Here s is a parameter defining the set $\mathcal{P}_{i.t}$. The set $\mathcal{P}_{i.t}$ is introduced in order to take into account the imprecision of $\mathbf{P}(i,t)$. It is assumed that sets $\mathcal{P}_{i.t}(s)$ are different for different i and t, and they are independent of each other. One of the well-known ways for dealing with imprecise data is to use the minimax or maximin (pessimistic or robust) strategy. In accordance with this strategy, a probability distribution is selected from every set of distributions $\mathcal{P}_{i.t}(s)$ such that the loss function achieves its largest value for fixed values of weights $\mathbf{w}$. It should be noted that the selected "optimal" probability distributions may be different for different values of weights $\mathbf{w}$. In fact, the minimax strategy selects the "worst" distribution providing the largest value of the loss function. Therefore, it can be interpreted as an insurance against the worst case because it aims at minimizing the expected loss in the least favorable case [31]. Robust models have been widely exploited in classification problems due to the opportunity to avoid some strong assumptions underlying the standard classification models [43]. Another "extreme" strategy is optimistic. It selects the "best" distribution providing the smallest value of the loss function. It can also be viewed as a direct opposite to the minimax strategy. However, the optimistic strategy cannot be interpreted as being robust.

Therefore, it is not studied below.

By applying the robust strategy, the problem (5) is written as the maximin optimization problem of the form:

$$\max_{\forall i,t:\mathbf{P}(i,t)\in\mathcal{P}_{i.t}(s)} \min_{\mathbf{w}\in\mathcal{W}(u)} \left(\sum_{i=1}^{N} \sum_{c=1}^{C} \left(\mathbf{p}_{i,c}\mathbf{w} - I_c(y_i)\right)^2 + \lambda \left\|\mathbf{w}\right\|^2 \right). \tag{6}$$

The maximization problem in (6) for every i and t is convex. Therefore, its solution can be found on bounds of $\mathcal{P}_{i.t}(s)$. Moreover, attempts to write a dual optimization problem in order to get the minimization problem lead to a non-linear optimization with quadratic constraints. This implies that the direct way for considering the imprecise relaxation of the RF class probability calculation and for computing the optimal weights cannot be applied in general, it can only be done for some simple cases of the classification problem statement.

Let us diverge from the standard definition of the loss function as the Euclidean distance between vectors $\mathbf{v}_i$ and $\mathbf{o}_i$ and consider an example of the weighted averaging for a RF consisting of $T = 3$ trees and solving a two-class classification problem ($C = 2$). Suppose that $y_i = 2$. This implies that the vector $\mathbf{o}_i$ is $(0, 1)$. Let us suppose that the output of the trees for example $\mathbf{x}_i$ are $p_1 = (0.1, 0.9)$, $p_2 = (0.6, 0.4)$, $p_3 = (0.3, 0.7)$. The weighted sum $v_{i,1} = 0.1w_1 + 0.6w_2 + 0.3w_3$ should be as close to 0 as possible, and the weighted sum $v_{i,2} = 0.9w_1 + 0.4w_2 + 0.7w_3$ should be as close to 1 as possible. It is important for us to make the second class probability $v_{i,2}$ close to 1. We concentrate only on this objective. In other words, it is proposed to consider only the weighted sum which corresponds to y_i and should be close to 1. Other weighted sums are not considered. Generalizing the above example, only probability $p_{y_i}(i, t)$ is used for the t-th tree and the i-th example. In fact, the dot product of vector $\mathbf{o}_i$ and the class probability distribution produced by the t-th decision tree is used. This means that the loss function given above is replaced with the following loss function:

$$\min_{\mathbf{w}\in\mathcal{W}(u)} \sum_{i=1}^{N} \left(1 - \sum_{t=1}^{T} p_{y_i}(i, t) w_t \right). \tag{7}$$

One can see from (7) that every term in the objective function contains only the weighted sum corresponding to y_i, i.e., to the unit element of the vector $\mathbf{o}_i$. It should be noted that the regularization term is not used here because it is assumed that its role is taken by the subset $\mathcal{W}(u)$. At the same time, a similar problem with the explicit regularization term will be studied later.

4.1 The linear programming problem

By taking into account (7), the maximin problem (6) can be written as follows:

$$N - \min_{\forall i,t:\mathbf{P}(i,t)\in\mathcal{P}_{i.t}(s)} \max_{\mathbf{w}\in\mathcal{W}(u)} \left(\sum_{i=1}^{N} \sum_{t=1}^{T} p_{y_i}(i, t) w_t \right),$$

which is equivalent to

$$\min_{\forall i,t:\mathbf{P}(i,t)\in\mathcal{P}_{i.t}(s)} \max_{\mathbf{w}\in\mathcal{W}(u)} \left(\sum_{t=1}^{T} w_t \sum_{i=1}^{N} p_{y_i}(i,t) \right). \tag{8}$$

Note that the sum $\sum_{i=1}^{N} p_{y_i}(i,t)$ divided by N is the mean probability that the t-th tree correctly classifies all examples from the training set. This implies that, by maximizing the objective function (8), one tries to find weights which lead to the largest mean probability of the correct classification.

Suppose that the set $\mathcal{W}(u)$ is produced by the following linear constraints:

$$a_t \le w_t \le b_t,\ t = 1, ..., T,\ \mathbf{w} \cdot \mathbf{1}^{\mathrm{T}} = 1. \tag{9}$$

These constraints correspond to most imprecise statistical models. Let us fix the variables $\mathbf{P}(i,t)$ and write the dual optimization problem for the primal form (8) with $\mathbf{w}$. It is of the form:

$$\min_{f_0,f_t,g_t} \left(f_0 + \sum_{t=1}^{T} (f_t b_t - g_t a_t) \right), \tag{10}$$

subject to $f_t, g_t \ge 0$, $t = 1, ..., T$,

$$f_0 + f_t - g_t \ge \sum_{i=1}^{N} p_{y_i}(i,t),\ t = 1, ..., T. \tag{11}$$

The dual problem is derived in the following standard way: each primal variable w_t becomes a dual constraint (11); every coefficient of a dual variable f_t or g_t in the dual constraint is the coefficient (1 or -1) of its primal variable in its primal constraint, respectively, that is f_t corresponds to the constraint $w_t \le b_t$, and g_t corresponds to the constraint $-w_t \le -a_t$; the right-hand side of every dual constraint is the primal objective function coefficient $\sum_{i=1}^{N} p_{y_i}(i,t)$; the optimization variable f_0 corresponds to the constraint $\mathbf{w} \cdot \mathbf{1}^{\mathrm{T}} = 1$.

Hence, there are two minimization problems (over $\mathbf{P}(i,t)$ and over f_0, f_c, g_c) which can be combined into a single problem, taking into account the problem with variables $\mathbf{P}(i,t)$. It is of the form:

$$\min_{\forall i,t:\mathbf{P}(i,t),\ f_0,f_t,g_t} \left(f_0 + \sum_{t=1}^{T} (f_t b_t - g_t a_t) \right), \tag{12}$$

subject to $f_t, g_t \ge 0$, $\mathbf{P}(i,t) \in \mathcal{P}_{i.t}(s)$, and (11).

So, a linear optimization problem has been obtained with $NTC + 2T + 1$ variables, namely $2T$ variables f_t and g_t, one variable f_0 and NTC variables $p_k(i,t)$. Since subsets $\mathcal{P}_{i.t}(s)$ are assumed to be different for all $t = 1, ..., T$ and $i = 1, ..., N$, the smallest value of $p_{y_i}(i,t)$ for every t and i is taken in order to provide the minimum of the objective function. This follows from the fact that the smallest values of $p_{y_i}(i,t)$ make the set of

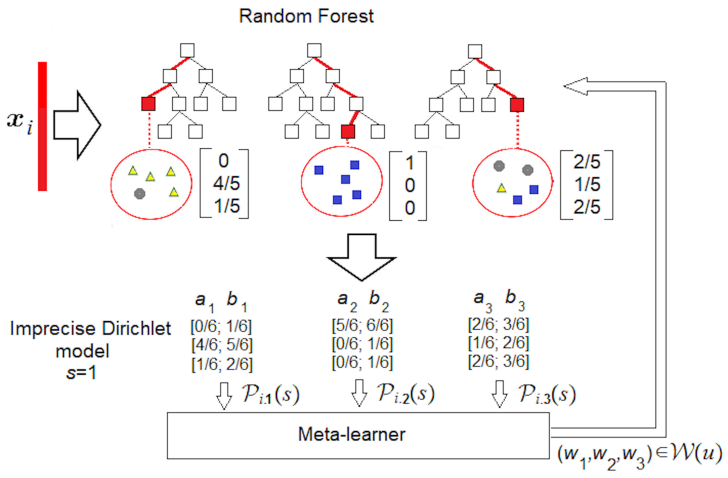

Figure 1: An example of the RF with a meta-learner using the IDM for the class probability interval-valued representation

feasible solutions to be larger. In other words, an extreme point of $\mathcal{P}_{i.t}(s)$ can be taken such that $p_{y_i}(i,t)$ is minimal. This implies that all extreme points of subset $\mathcal{P}_{i.t}(s)$ have to be found. It should be noted that several extreme points may have identical values of $p_{y_i}(i,t)$. It does not matter because only one probability $p_{y_i}(i,t)$ from all probabilities of the class probability distribution $\mathbf{P}(i,t)$ is used for every i and t.

Let us denote the smallest values of $p_{y_i}(i,t)$ as $p^*_{y_i}(i,t)$. In summary, the smallest probabilities $p^*_{y_i}(i,t)$ have to be found among elements of extreme points. The probabilities do not depend on other variables f_0, f_c, g_c. Therefore, one can return to the primal form (8) and use the smallest values $p^*_{y_i}(i,t)$ in (8). Hence, optimal weights are computed from the linear optimization problem (8) by using the extreme points of $\mathcal{P}_{i.t}(s)$. It should be noted that the dual problem (10)-(11) or problem (12) is needed in order to simply show how to find optimal values of $\mathbf{P}(i,t)$ from $\mathcal{P}_{i.t}(s)$.

An example of the RF consisting of three decision trees with a meta-learner which computes a vector of optimal weights (w_1, w_2, w_3) is shown in Fig. 1. Precise probabilities of three-class examples which fall into shaded leaves are $(0, 4/5, 1/5)$, $(1, 0, 0)$, $(2/5, 1/5, 2/5)$. Examples from different classes are denoted by circles, triangles and squares. Intervals with bounds a_i and b_i obtained by using the IDM with the hyperparameter $s = 1$ are also shown in Fig. 1. By using these intervals, sets $\mathcal{P}_{i.t}(s)$ for every tree and every example are constructed, and the meta-learner solves the maximin optimization problem in order to compute optimal weights which are assigned to trees to classify test data.

4.2 The quadratic programming problem

So far, it has been studied how to compute optimal weights of trees under the condition that constraints for weights play a role for the regularization. Let us now consider the explicit regularization $\|\mathbf{w}\|^2$ added to the set $\mathcal{W}(u)$. In this case, the following quadratic programming problem is obtained:

$$\max_{\forall i,t:\mathbf{P}(i,t)\in\mathcal{P}_{i.t}(s)} \min_{\mathbf{w}\in\mathcal{W}(u)} \left(\lambda \|\mathbf{w}\|^2 - \sum_{t=1}^{T} w_t \sum_{i=1}^{N} p_{y_i}(i,t) \right). \tag{13}$$

If one fixes the variables $\mathbf{P}(i,t)$, then a standard convex quadratic programming problem with linear constraints with respect to $\mathbf{w}$ is obtained. The problem (13) can be viewed as an extension of (5). Let us again find the dual problem in order to prove that the optimal solution for probability distribution $\mathbf{P}(i,t)$ should be found among the largest elements with the index y_i. The dual problem for the minimization problem jointly with the optimization problem over $\mathbf{P}(i,t)$ can be written as

$$\max_{\forall i,t:\mathbf{P}(i,t)\in\mathcal{P}_{i.t}(s)} \max_{\mathbf{v},f_0,f_t,g_t} \left(-\lambda \|\mathbf{v}\|^2 - f_0 - \sum_{t=1}^{T} (f_t b_t - g_t a_t) \right),$$

subject to $f_t, g_t \geq 0$, $t = 1, ..., T$, and

$$f_0 + f_t - g_t + 2\lambda v_t \geq \sum_{i=1}^{N} p_{y_i}(i,t),\ t = 1, ..., T.$$

Here $\mathbf{v} = (v_1, ..., v_T)$ is a vector of T slack variables, f_t, g_t are non-negative optimization variables, $t = 1, ..., T$, f_0 is the optimization variable.

The dual form is not considered in detail because it is obtained by means of a standard formal procedure. It is important for us to see that the maximum of the objective function is achieved when sums $\sum_{i=1}^{N} p_{y_i}(i,t)$ are minimal. Hence, the smallest values $p^*_{y_i}(i,t)$ are substituted into (13), and the following standard quadratic optimization problem for computing optimal weights is solved:

$$\min_{\mathbf{w}\in\mathcal{W}(u)} \left(\lambda \|\mathbf{w}\|^2 - \sum_{t=1}^{T} w_t \sum_{i=1}^{N} p^*_{y_i}(i,t) \right). \tag{14}$$

Problem (14) is a result of the double regularization. First, the regularization term $\|\mathbf{w}\|^2$ is used to restrict the set of weights. Secondly, the weights are also restricted by the set $\mathcal{W}(u)$.

4.3 A special case: the IDM for class probabilities and the linear-vacuous mixture for weights

Let us consider the *IDM* [40] and the *linear-vacuous mixture* or *imprecise ε-contaminated models* [39] as special cases of models for defining subsets $\mathcal{P}_{i.t}(s)$ and subset $\mathcal{W}(u)$, respectively.

The IDM is defined by [40] as the set of all Dirichlet distributions over $\pi = (\pi_1, ..., \pi_T)$ with parameters $\alpha = (\alpha_1, ..., \alpha_T)$ and s such that the vector α belongs to the unit simplex and every α_i is the mean of π_i under the Dirichlet prior. Here π is the probability distribution defined on T events, for which the Dirichlet (s, α) prior is defined. For the IDM, the hyperparameter s determines how quickly upper and lower probabilities of events converge as statistical data accumulate. Smaller values of s produce faster convergence and stronger conclusions, whereas large values of s produce more cautious inferences. Detailed discussion concerning the parameter s and the IDM can be found in [8, 40]. In the framework of classification, the hyperparameter s can be regarded as a tuning parameter. Therefore, its optimal value can be obtained only by means of using the cross-valudation procedure and comparison of the classification or regression results. At the same time, the value of s must not depend on the number of observations.

Let A be any non-trivial subset of a sample space $\{\vartheta_1, ..., \vartheta_T\}$, and let $n(A)$ denote the observed number of occurrences of A in the N trials, $n(A) = \sum_{\vartheta_j \in A} n_j$. Then, according to [40], the predictive probability $P(A, s)$ under the Dirichlet posterior distribution is

$$P(A, s) = \frac{n(A) + s\alpha(A)}{N + s},$$

where $\alpha(A) = \sum_{\vartheta_j \in A} \alpha_j$.

By maximizing and minimizing α_j under restriction $\alpha \in S(1, m)$, the posterior upper and lower probabilities of A are obtained:

$$\underline{P}(A, s) = \frac{n(A)}{N + s}, \ \overline{P}(A, s) = \frac{n(A) + s}{N + s}.$$

We return to the definition of the tree class probability assuming that the i-th training example $(\mathbf{x}_i, y_i)$ falls into a leaf node with number $l(t)$ of the t-th decision tree of a RF. If the vector of stored votes corresponding to this leaf node is $\mathbf{n}_{l(t)} = (n_{l(t),1}, ..., n_{l(t),C})$, then one can find bounds for probability $p_{l(t),c}$ in accordance with the IDM. It follows from the definition of the IDM that the bounds for the probability are of the form:

$$a_{l(t),c} = \frac{n_{l(t),c}}{n_{l(t)} + s} \leq p_{l(t),c} \leq \frac{n_{l(t),c} + s}{n_{l(t)} + s} = b_{l(t),c}.$$

The subset of probability distributions $\mathcal{P}_{i.t}(s)$ has C extreme points $(q_1(t), ..., q_C(t))$ such that the c-th extreme point, $c = 1, ..., C$, is determined in a simple way as follows:

$$q_k(t) = b_{l(t),c}, \ q_c(t) = a_{l(t),c}, \ c = 1, ..., C, \ c \neq k.$$

It is obvious that the smallest probability $p^*_{y_i}(i,t)$ is equal to $a_{l(t),c}$.

Let us suppose that the set $\mathcal{W}(u)$ is produced by means of the linear-vacuous mixture or the imprecise ε-contaminated models [39] with the elicited probability distribution $(T^{-1},...,T^{-1})$ and the parameter $\varepsilon \in [0,1]$, i.e., $u = \varepsilon$. Generally, the elicited probability distribution can be arbitrary in the unit simplex. If $\varepsilon = 0$, then the set $\mathcal{W}(u)$ is reduced to the point $(T^{-1},...,T^{-1})$. The set $\mathcal{W}(u)$ coincides with the unit simplex when $\varepsilon = 0$. According to the model, $\mathcal{W}(u)$ is the convex set of weights with lower bound $(1-\varepsilon)T^{-1}$ and upper bound $(1-\varepsilon)T^{-1}+\varepsilon$, i.e.,

$$(1-\varepsilon)T^{-1} \le w_t \le (1-\varepsilon)T^{-1}+\varepsilon,\ t = 1,...,T.$$

This model can be viewed as another form of the IDM. In particular, there is a connection between parameters s and ε, which is $\varepsilon = s/(T+s)$. It has T extreme points denoted as $\mathcal{E}(\mathcal{W}(\varepsilon))$, which are all of the same form: the k-th element is given by $(1-\varepsilon)T^{-1}+\varepsilon$ and the other $T-1$ elements are equal to $(1-\varepsilon)T^{-1}$, i.e.,

$$q_k(t) = \frac{(1-\varepsilon)}{T}+\varepsilon,\ q_t(t) = \frac{(1-\varepsilon)}{T},\ t = 1,...,T,\ t \ne k.$$

Let us denote

$$A_t = \sum_{i=1}^{N} a_{l(t),y_i} = \sum_{i=1}^{N} \frac{n_{l(t),y_i}}{n_{l(t)}+s}. \tag{15}$$

Then problem (8) can be rewritten by taking into account the extreme points of $\mathcal{W}(\varepsilon)$ as

$$\max_{\mathbf{w}\in\mathcal{E}(\mathcal{W}(\varepsilon))} \sum_{t=1}^{T} w_t A_t = \sum_{t=1}^{T} \frac{(1-\varepsilon)}{T} A_t + \varepsilon \max_{k=1,...,T} A_k.$$

The above implies that the optimal weight vector consists of $T-1$ elements $(1-\varepsilon)T^{-1}$ and one element $(1-\varepsilon)T^{-1}+\varepsilon$. At that, the tree, which provides the largest value of A_t, is assigned by the weight $(1-\varepsilon)T^{-1}+\varepsilon$. This solution is trivial and does not take into account a difference between trees, except for one tree with the largest value of A_k. It is given here as an example. The quadratic problem (14) solves this problem and provides better results. By taking into account the introduced parameters A_t in (15), problem (14) can be rewritten as follows:

$$\min_{\mathbf{w}} \left(\lambda \left\| \mathbf{w} \right\|^2 - \sum_{t=1}^{T} A_t w_t \right),$$

subject to

$$\frac{(1-\varepsilon)}{T} \le w_t \le \frac{(1-\varepsilon)}{T}+\varepsilon,\ t = 1,...,T,\ \mathbf{w}\cdot\mathbf{1}^{\mathrm{T}} = 1. \tag{16}$$

5 Weighted RFs for Regression

5.1 Weighted Averages

Let us formally state the standard regression problem. Given N training data (examples, instances, patterns) $S = \{(\mathbf{x}_1, y_1), ..., (\mathbf{x}_N, y_N)\}$, in which $\mathbf{x}_i$ may belong to an arbitrary set $\mathcal{X} \subset \mathbb{R}^m$ and represents a feature vector involving m features, and $y_i \in \mathbb{R}$ represents the observed output such that $y_i = f(\mathbf{x}_i) + \varepsilon$. Here ε is the random noise with expectation 0 and finite variance. Machine learning aims to construct a regression model $f(\mathbf{x})$ that minimizes the expected risk which, for example, can be represented by the least squares error criterion

$$J = \frac{1}{N} \sum_{i=1}^{N} \left(y_i - f(\mathbf{x}_i) \right)^2 . \tag{17}$$

A powerful nonparametric statistical method for the regression problems is the regression RF. Before considering the whole RF, it can be defined how predicted values can be computed by a single decision tree. Suppose that n_l training examples $(\mathbf{x}_i, y_i)$ with indices from a set D_l, i.e., $i \in D_l$, fall into the l-th leaf node. Then an output $z = f(\mathbf{x})$ of a new example which falls into the l-th leaf node can be computed as follows:

$$z = f(\mathbf{x}) = \frac{1}{n_l} \sum_{i \in D_l} y_i .$$

In other words, the predicted value z assigned to the l-th leaf is the average of outputs of all examples with indices from D_l. This implies that the predicted value strongly depends on the number of examples whose indices belong to D_l. If this number is small, then it is quite unreasonable to expect that the value z will be correct.

Let us return to the RF and suppose that it consists of T trained trees. The RF output z_{RF} is computed by averaging all predicted values $z^{(t)}$ across all trees, i.e., there holds

$$z_{\mathrm{RF}} = \frac{1}{T} \sum_{t=1}^{T} z^{(t)} .$$

In order to improve RFs, weights w_t can be assigned to decision trees and the weighted average of the tree predictions is computed as

$$z_{\mathrm{RF}} = \sum_{t=1}^{T} z^{(t)} w_t = \mathbf{w}\mathbf{z} .$$

Here $\mathbf{w} = (w_1, ..., w_T)$ is the weight vector, $\mathbf{z} = (z^{(1)}, ..., z^{(T)})^{\mathrm{T}}$ is the vector of T tree outputs corresponding to example $\mathbf{x}$. The weights are also restricted by condition (3).

By using the weights, the least squares error criterion (17) can be rewritten as follows:

$$J(\mathbf{w}) = \frac{1}{N} \sum_{i=1}^{N} \left(y_i - \mathbf{w}\mathbf{z}_i \right)^2 ,$$

where $\mathbf{z}_i$ is the vector of T tree outputs corresponding to training example $\mathbf{x}_i$.

It is obvious that estimates of z for every leaf cannot be considered to be precise when based on a small number of training data. This implies that interval-valued predicted values should be taken in place of precise ones. Using standard statistical estimation techniques, a confidence interval is obtained as:

$$z \pm t_{(1-a/2)} s_z / \sqrt{n_l}, \tag{18}$$

where $t_{(1-a/2)}$ is the percentile from the t-distribution corresponding to a $100(1-\alpha)$ percent confidence interval for the mean z value or the t-values;

$$s_z^2 = \frac{1}{n_l - 1} \sum_{i \in D_l} (y_i - z)^2.$$

Hence, the value $z^{(t)}$ for the t-th tree has lower and upper bounds, $z_L^{(t)}$ and $z_U^{(t)}$, respectively, which have to be taken into account in the problem of computing the weights. Let $\mathcal{Z}^{(t)}$ be a set of $z^{(t)}$ obtained from interval $[z_L^{(t)}, z_U^{(t)}]$. Since every $z^{(t)}$ depends on the specific example as well as on the tree, the t-th output of the i-th example will be denoted as $z_i^{(t)}$ and the corresponding interval as $\mathcal{Z}_i^{(t)}$.

The pessimistic or robust strategy [31] is again used. In accordance with this strategy, the "worst" values of $z^{(t)}$ are selected from intervals $\mathcal{Z}^{(t)}$, $t = 1, ..., T$, providing the largest value of the loss function $J(\mathbf{w})$.

5.2 Training the Weighted Regression RF

In order to find optimal weights $\mathbf{w}$ by known precise values of $z_i^{(t)}$, an optimization problem has to be solved taking into account that vectors $\mathbf{v}_i$ will be as close as possible to vectors $\mathbf{o}_i$ whose unit element has the index y_i coinciding with the class label of $\mathbf{x}_i$. It can be done by minimizing the loss function which is defined by the distance $d(\mathbf{v}_i, \mathbf{o}_i)$ between $\mathbf{v}_i$ and $\mathbf{o}_i$, i.e.,

$$\min_{\mathbf{w}} \left(\sum_{i=1}^{N} (y_i - \mathbf{w}\mathbf{z}_i)^2 + \lambda \|\mathbf{w}\|^2 \right).$$

Here $\|\mathbf{w}\|^2$ is a regularization term and λ is a hyper-parameter which controls the strength of the regularization.

The set of weights is also restricted by some convex subset $\mathcal{W}(u)$ of the unit simplex in order to improve the regularization. Here u is another regularization parameter which defines the size of the subset $\mathcal{W}$ (see the description of the same parameter and the set $\mathcal{W}(u)$ for the classification problem).

In order to take into account that $z_i^{(t)}$ is interval-valued and to apply the robust strategy, the maximin optimization problem is written as

$$\max_{z_i^{(t)} \in \mathcal{Z}_i^{(t)}} \min_{\mathbf{w} \in \mathcal{W}(u)} \left(\sum_{i=1}^{N} (y_i - \mathbf{w}\mathbf{z}_i)^2 + \lambda \|\mathbf{w}\|^2 \right).$$

Unfortunately, the above representation cannot be applied because it leads to a quadratic optimization problem with quadratic constraints. Therefore, the first term of the objective function is replaced with

$$\sum_{i=1}^{N}\left|y_i-\sum_{t=1}^{T}z_i^{(t)}w_t\right|=\sum_{i=1}^{N}\left|\sum_{t=1}^{T}w_t\left(y_i-z_i^{(t)}\right)\right|,$$

and then

$$\sum_{t=1}^{T}w_t\sum_{i=1}^{N}\left|y_i-z_i^{(t)}\right|.$$

The last transformation, of course, is not equivalent. However, in fact, it is proposed to replace the first loss function by another loss function which is stronger because it restricts the classification error for every tree, but not for the whole RF. This leads to the following quadratic optimization problem is obtained:

$$\max_{z_i^{(t)}\in\mathcal{Z}_i^{(t)}}\min_{\mathbf{w}\in\mathcal{W}(u)}\left(\sum_{t=1}^{T}w_t\sum_{i=1}^{N}\left|\left(y_i-z_i^{(t)}\right)\right|+\lambda\left\|\mathbf{w}\right\|^2\right). \tag{19}$$

This is the primal form of the optimization problem. Let us fix the value $z_i^{(t)}$ and write the dual optimization problem for $\mathbf{w}$. Suppose again that the set $\mathcal{W}(u)$ is produced by constraints (9) which correspond to most imprecise statistical models. The dual problem is written without the regularization term for simplicity. Jointly with optimization over $z_i^{(t)}$, it is of the form:

$$\max_{z_i^{(t)}\in\mathcal{Z}_i^{(t)}}\max_{h_0,h_t,g_t}\left(h_0+\sum_{t=1}^{T}\left(h_tb_t-g_ta_t\right)\right), \tag{20}$$

subject to $h_t,g_t\geq 0$, $t=1,...,T$,

$$h_0+h_t-g_t\leq\sum_{i=1}^{N}\left|\left(y_i-z_i^{(t)}\right)\right|,\ t=1,...,T. \tag{21}$$

Here h_t,g_t are non-negative optimization variables, $t=1,...,T$, h_0 is the optimization variable.

It can be seen from the above optimization problem that the right sides of the constraints have to be maximized to get the maximum over $z_i^{(t)}$, which is achieved when the following rule holds: if $y_i<(z_{i,L}^{(t)}+z_{i,U}^{(t)})/2$, then $z_i^{(t)*}=z_{i,U}^{(t)}$, otherwise $z_i^{(t)*}=z_{i,L}^{(t)}$. Here $z_i^{(t)*}$ is an optimal value of the $z_i^{(t)}$ from the interval $\mathcal{Z}_i^{(t)}$ maximizing the objective functions (19) or (20).

The same results can be obtained for the quadratic optimization problem (19) which takes into account the regularization term $\lambda\left\|\mathbf{w}\right\|^2$. The dual problem is of the form:

$$\max_{z_i^{(t)}\in\mathcal{Z}_i^{(t)}}\max_{v,h_0,h_t,g_t}\left(-\lambda\left\|\mathbf{v}\right\|^2-h_0+\sum_{t=1}^{T}\left(h_tb_t-g_ta_t\right)\right), \tag{22}$$

subject to $h_t, g_t \geq 0$, $t = 1, ..., T$,

$$h_0 + (h_t - g_t) + 2\lambda v_t \geq -\sum_{i=1}^{N} \left| \left(y_i - z_i^{(t)} \right) \right|, \ t = 1, ..., T. \tag{23}$$

Hence, the corresponding optimal values of $z_i^{(t)*}$ depending on y_i, $z_{i,L}^{(t)}$, $z_{i,U}^{(t)}$ are substituted into the primal form (19), and the following standard quadratic optimization problem for computing optimal weights is solved:

$$\min_{\mathbf{w} \in \mathcal{W}(u)} \left(\sum_{t=1}^{T} w_t \sum_{i=1}^{N} \left| \left(y_i - z_i^{(t)*} \right) \right| + \lambda \left\| \mathbf{w} \right\|^2 \right). \tag{24}$$

Let us suppose that the set $\mathcal{W}(u)$ is produced by means of the imprecise ε-contaminated model [39] with the elicited probability distribution $(T^{-1}, ..., T^{-1})$ and the parameter $\varepsilon \in [0, 1]$, i.e., $u = \varepsilon$. According to this model, the set $\mathcal{W}(\varepsilon)$ is produced by T constraints (16). Then the quadratic optimization problem has linear constraints and can be solved using standard procedures.

6 Numerical experiments

6.1 Classification

In order to illustrate the robust classification RF model, it is compared to the original RF by using datasets from UCI Machine Learning Repository [22]. Table 1 shows the number of features m for the corresponding data set, the number of examples N and the number of classes C. More detailed information can be found from the data resources.

The accuracy measure A used in the classification experiments is the proportion of correctly classified cases on a sample of data. To evaluate the average accuracy, a cross-validation with 100 repetitions is performed, where in each run, N_{tr} training data and $N_{\text{test}} = N - N_{\text{tr}}$ testing data are randomly selected. Every RF consists of 100 decision trees.

The preliminary numerical experiments have shown that a large number of weights may lead to overfitting. In order to overcome this difficulty and to reduce the complex optimization problems, it is proposed to unite all trees into G groups. The class probability distribution for every group is determined by averaging all class probability distributions in the group. The number of groups is a tuning parameter of the machine learning task. It is impossible to define an optimal number before numerical experiments. Therefore, it is done empirically by taking different numbers from the set G and comparing the validation results.

For experiments, the following values of N_{tr} are considered: 10, 20, 30, 40. Different values for the model hyper-parameters have been tested, choosing those leading to the best results. They are selected from the following sets of the hyper-parameters:

Table 1: A brief introduction about data sets for classification

Data set	Abbreviation	m	N	C
QSAR Biodegradation	Biodeg	41	1055	2
Cardiotocography	Cardio3	21	2126	3
Diabetic Retinopathy	Diabet	19	1151	2
Haberman's Breast Cancer Survival	HS	3	306	2
Ionosphere	Ion	34	351	2
Parkinsons	Parkinsons	22	195	2
Website Phishing	Phishing_1	9	1353	3
Seeds	Seeds	7	210	3
Seismic-Bumps	Seismic	24	2584	2
Connectionist Bench	Sonar	60	208	2
SPECT Heart	SPECT	22	267	2
SPECTF Heart	SPECTF	44	267	2
Teaching Assistant Evaluation	TAE	54	151	3
Tic-Tac-Toe Endgame	TTTE	27	958	2
Wholesale Customer	WC3	6	440	3
Breast Cancer Wisconsin (Diagnostic)	WDBC	30	569	2

- The regularization hyper-parameter $\lambda \in \{0.5, 1, 1.5, ..., 10, 10.5\}$.
- The contamination parameter $\varepsilon \in \{0.25, 0.5, 0.75\}$.
- The parameter of the number of the tree groups $G \in \{5, 10, 20, 25, 100\}$.
- The hyper-parameter of the IDM $s \in \{1, 3, 5\}$.

It should be noted that all results are obtained for three values of s.

The numerical results for cases using 10, 20, 30, 40 training examples are shown in Tables 2-5, respectively. The best performance for each dataset is shown in bold. It can be seen from Tables 2-5 that the IRF outperforms the original RF for all datasets. Table 6 illustrates the mean values of the accuracy measures across all datasets for cases using 10, 20, 30, 40 training examples. It follows from Table 6 that the mean accuracy measures of the IRF are larger than the same measures of the original RF.

In order to formally show when the proposed IRF outperforms the original RF, the t-test is applied, which has been proposed and described by Demsar [12] for testing whether the average difference in the performance of two classifiers, the IRF and the RF, is significantly different from zero. The largest value of the IRF accuracy measures for $s = 1, 3, 5$ are

Table 2: Comparison of the RF with the IRF by $s = 1, 3, 5$ when 10 examples are used for training

Data set	RF	IRF, $s = 1$	IRF, $s = 3$	IRF, $s = 5$
Biodeg	71.34	**74.48**	74.13	74.04
Cardio3	78.64	**80.47**	80.12	79.86
Diabet	54.22	**58.53**	58.34	58.12
HS	72.69	75.03	**75.07**	74.92
Ion	72.80	77.55	**77.61**	77.49
Parkinsons	79.20	**82.43**	82.07	81.72
Phishing_1	76.76	**79.89**	79.50	79.41
Seeds	84.31	**87.19**	86.86	86.43
Seismic	92.93	**93.07**	93.06	93.05
Sonar	65.28	**70.18**	69.85	69.50
SPECT	79.08	**79.40**	79.22	79.16
SPECTF	78.78	**80.47**	80.04	79.99
TAE	35.81	**40.77**	39.96	39.55
TTTE	64.70	**67.59**	67.36	67.32
WC3	70.39	71.70	71.74	**71.85**
WDBC	90.45	92.80	92.70	**92.36**

Table 3: Comparison of the RF with the IRF by $s = 1, 3, 5$ when 20 examples are used for training

Data set	RF	IRF, $s = 1$	IRF, $s = 3$	IRF, $s = 5$
Biodeg	73.54	**76.83**	76.64	76.37
Cardio3	79.16	**82.20**	82.07	81.78
Diabet	56.10	**59.77**	59.6	59.57
HS	73.85	75.33	**75.49**	75.48
Ion	75.18	80.39	80.36	**80.39**
Parkinsons	81.63	84.43	**84.45**	84.37
Phishing_1	80.40	**83.14**	83.04	83.05
Seeds	86.67	89.56	89.35	89.02
Seismic	93.00	**93.10**	93.05	93.04
Sonar	69.82	**73.61**	73.48	73.27
SPECT	79.00	**79.18**	79.10	79.14
SPECTF	79.01	**80.49**	80.38	80.29
TAE	38.47	**43.74**	43.50	43.21
TTTE	65.67	**69.61**	69.35	69.08
WC3	71.02	**71.76**	71.66	71.65
WDBC	91.51	**93.37**	93.26	93.24

Table 4: Comparison of the RF with the IRF by $s = 1, 3, 5$ when 30 examples are used for training

Data set	RF	IRF, $s = 1$	IRF, $s = 3$	IRF, $s = 5$
Biodeg	74.08	**77.31**	77.19	76.97
Cardio3	79.11	**82.06**	81.97	81.72
Diabet	57.55	**61.47**	61.37	61.30
HS	73.84	**75.44**	75.36	75.40
Ion	79.28	**83.06**	82.96	83.03
Parkinsons	82.26	85.03	**85.09**	85.08
Phishing_1	80.92	82.84	**82.85**	82.82
Seeds	87.46	**89.79**	89.71	89.58
Seismic	92.99	**93.10**	93.08	93.08
Sonar	72.73	**76.38**	76.30	76.16
SPECT	79.00	**79.16**	79.09	79.13
SPECTF	79.09	80.34	**80.39**	80.28
TAE	38.27	**44.16**	44.06	43.50
TTTE	65.60	**69.47**	69.45	69.35
WC3	71.62	71.93	**71.94**	**71.94**
WDBC	91.69	**93.43**	93.42	93.24

Table 5: Comparison of the RF with the IRF by $s = 1, 3, 5$ when 40 examples are used for training

Data set	RF	IRF, $s = 1$	IRF, $s = 3$	IRF, $s = 5$
Biodeg	73.22	**77.16**	76.86	76.54
Cardio3	79.25	**82.71**	82.59	82.45
Diabet	58.22	61.74	**61.84**	61.82
HS	73.85	**75.55**	75.44	75.37
Ion	79.37	**84.31**	83.86	83.88
Parkinsons	82.24	84.67	84.77	**84.86**
Phishing_1	81.68	**83.42**	83.36	83.31
Seeds	87.44	**89.60**	89.53	89.35
Seismic	92.99	**93.15**	93.13	93.11
Sonar	73.28	**76.79**	76.67	76.76
SPECT	79.00	79.03	**79.04**	79.03
SPECTF	79.01	**80.61**	80.41	80.29
TAE	40.64	**46.30**	46.23	46.03
TTTE	65.44	70.16	70.32	**70.33**
WC3	71.88	72.09	72.10	**72.11**
WDBC	92.32	**93.91**	93.90	93.83

Table 6: Average accuracy measures for the RF with the IRF by $s = 1, 3, 5$ across all datasets

	RF	IRF, $s = 1$	IRF, $s = 3$	IRF, $s = 5$
10	72.96	**75.72**	75.48	75.30
20	74.63	**77.28**	77.17	77.06
30	75.34	**77.81**	77.76	77.66
40	75.61	**78.20**	78.13	78.07

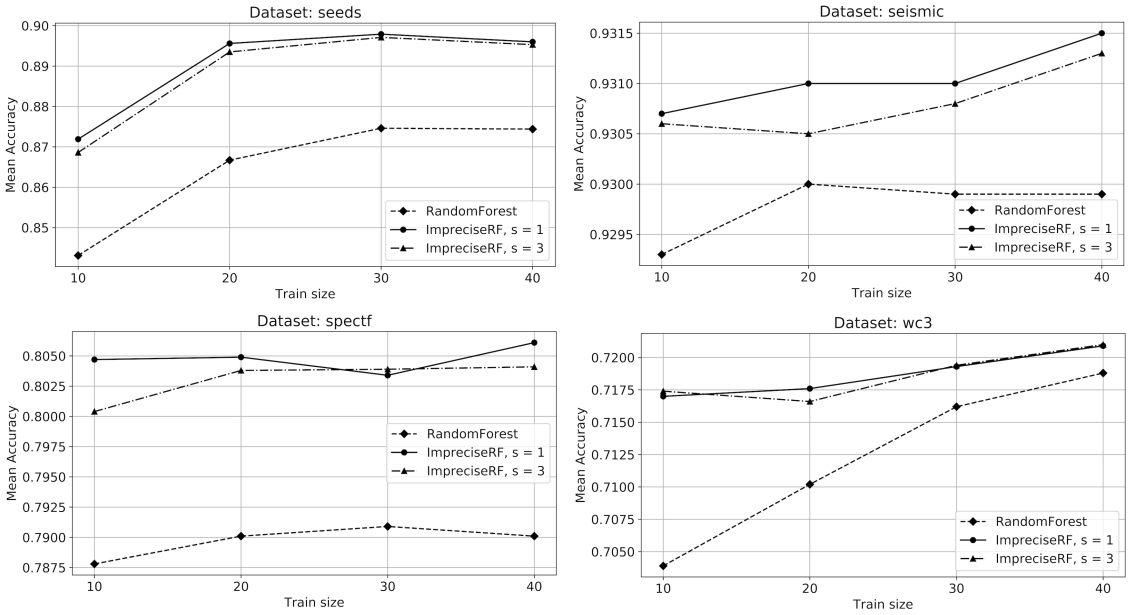

Figure 2: Illustration of the proposed model outperformance for datasets: Seeds, Seismic, Spectf, WC3

compared with the corresponding RF accuracy measure. The t statistic in this case is distributed according to the Student's t-distribution with $16 - 1$ degrees of freedom. The t statistics for the differences lead to the p-values. They are 2.1×10^{-6}, 6.4×10^{-6}, 1.7×10^{-5}, 2.7×10^{-5} for cases with 10, 20, 30, 40 training examples, respectively. It can be seen from the results that the proposed model outperforms the original RF for all cases, but its relative level of improved performance compared to the original RF is reduced with increase of the number of training examples.

Fig. 2 illustrates how the accuracy measures depend on the training size for datasets: Seeds, Seismic, Spectf, WC3. The lines with diamond markers, circle markers, and triangle markers correspond to the original RF, the IRF by $s = 1$, and the IRF by $s = 3$, respectively. Such pictures are not provided for all datasets because they are very similar, the corresponding accuracy measures are given in Tables 2-5.

Table 7: Comparison of the SVM and the RotForest with the IRF when 40 examples are used for training

Data set	SVM	RotForest	IRF
Biodeg	72.09	74.18	**77.16**
Cardio3	76.88	79.32	**82.71**
Diabet	58.22	58.22	**61.84**
HS	74.38	73.06	**75.55**
Ion	83.20	80.65	**84.31**
Parkinsons	84.04	83.72	**84.86**

Table 8: Comparison of the RF with the IRF by $s = 1, 3, 5$ when 10 examples are used for training by noisy testing data

Data set	RF	IRF, $s = 1$	IRF, $s = 3$	IRF, $s = 5$
Cardio3	78.22	**79.69**	79.14	78.96
HS	71.94	74.11	74.30	**74.45**
Ion	69.49	**74.12**	73.82	73.62
Seeds	77.84	**81.01**	80.36	80.02
Sonar	64.53	**68.71**	**68.71**	68.53
SPECTF	78.78	**80.33**	80.30	80.19
TTTE	64.99	**67.89**	67.49	67.21
WC3	69.25	71.36	71.62	**71.79**

To compare the proposed IRF with other classification models, we used the well-known support vector machine (SVM) and the Rotation Forest (RotForest) [32]. An implementation of RotForest in Phyton can be found at https://github.com/antongoy/RotationAlgorithms. The SVM uses a standard Gaussian radial basis function (RBF) kernel in all experiments. Different values for the kernel parameter and the "cost" parameter of the SVM have been tested, choosing those leading to the best results. The largest values of A obtained in the previous experiments are taken for the IRF. Results of experiments on some datasets are shown in Table 7. It can be seen from Table 7 that the SVM and the RotForest provide worse results in comparison with the IRF for the small datasets of size 40.

In order to study the robustness of the IRF, noise is added to the testing data in the following way. For every feature j, its standard deviation σ_j using the whole dataset is computed. If x_{ij} is the j-th feature of the i-th example from the testing set, then the corresponding feature with the noise becomes $x_{ij} + z_{ij}$, where z_{ij} is a normally distributed random number with zero expectation and standard deviation $\eta\sigma_j$. Here η is a parameter which is set as 0.5 in experiments. The corresponding results for cases with 10, 20, 30, 40 training examples are shown in Tables 8-11, respectively. One can again observe that the IRF outperforms the original RF for all datasets.

Table 9: Comparison of the RF with the IRF by $s = 1, 3, 5$ when 20 examples are used for training by noisy testing data

Data set	RF	IRF, $s = 1$	IRF, $s = 3$	IRF, $s = 5$
Cardio3	78.60	**81.01**	80.99	80.91
HS	73.60	**75.08**	74.95	74.97
Ion	70.94	76.78	**76.82**	76.79
Seeds	81.24	**83.36**	**83.36**	83.17
Sonar	67.94	71.59	**71.63**	71.58
SPECTF	78.87	**80.43**	80.42	80.40
TTTE	65.86	68.70	**68.82**	68.74
WC3	70.67	**71.78**	71.76	71.72

Table 10: Comparison of the RF with the IRF by $s = 1, 3, 5$ when 30 examples are used for training by noisy testing data

Data set	RF	IRF, $s = 1$	IRF, $s = 3$	IRF, $s = 5$
Cardio3	78.30	**80.58**	80.51	80.49
HS	73.46	75.07	**75.27**	75.22
Ion	73.78	**79.46**	79.33	79.33
Seeds	81.09	**83.86**	83.78	83.71
Sonar	69.47	**72.96**	72.88	72.91
SPECTF	78.91	80.50	80.50	**80.52**
TTTE	65.21	**68.49**	68.42	68.32
WC3	71.32	**71.97**	71.91	71.90

Fig. 3 illustrates how the accuracy measures depend on the size of the training set by adding the noise to the testing data for datasets: Cardio3, Seeds, Spectf, WC3. The lines with diamond markers, circle markers, and triangle markers correspond to the original RF, the IRF by $s = 1$, and the IRF by $s = 3$, respectively.

Deriving the proposed IRF differs from deriving the original RF classifier by additional computation of the extreme points and solving the quadratic optimization problem in the training phase. The extreme points are known in explicit form. Therefore, their computation practically does not require additional time. The quadratic optimization problem is also simple because the number of training examples is small. When a PC with Intel Core i7-7700HQ 2.80GHz is used, the IRF solution, compared to the RF solution, requires the additional time for training, which is less than one second. The computation times for the predictions for both methods do not show any difference, because the standard averaging operation for RF is just replaced by weighted averaging for IRF.

Table 11: Comparison of the RF with the IRF by $s = 1, 3, 5$ when 40 examples are used for training by noisy testing data

Data set	RF	IRF, $s = 1$	IRF, $s = 3$	IRF, $s = 5$
Cardio3	78.39	**81.14**	81.10	81.04
HS	74.00	**75.49**	75.39	75.38
Ion	73.43	79.28	79.26	**79.30**
Seeds	80.70	83.21	**83.33**	83.22
Sonar	70.76	74.02	73.99	**74.09**
SPECTF	79.00	80.69	**80.77**	80.70
TTTE	65.51	69.53	**69.54**	69.44
WC3	71.53	**71.96**	71.92	71.88

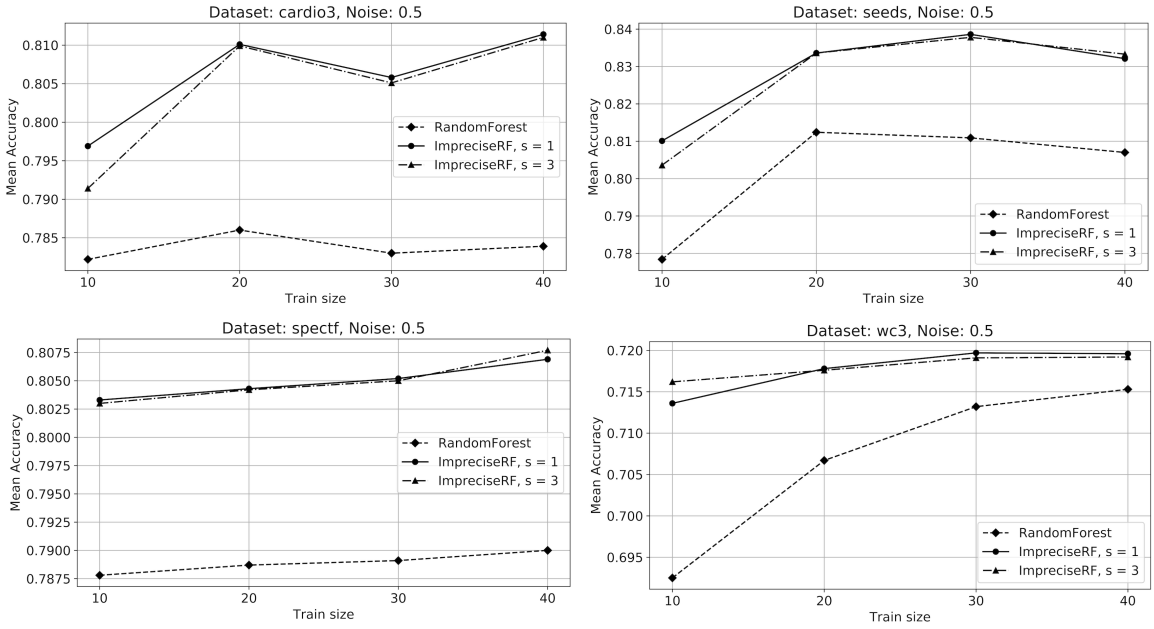

Figure 3: Illustration of the proposed model outperformance for datasets: Cardio3, Seeds, Spectf, WC3

Table 12: A brief introduction about data sets for regression

Data set	Abbreviation	m	N
Airfoil Self-Noise	Airfoil	6	1503
Concrete Compressive Strength	Concrete	9	1030
Energy Efficiency: Cooling Load	Energy_c	8	768
Energy Efficiency: Heating Load	Energy_h	8	768
Real Estate Valuation Data	Estate	7	414
Physicochemical Properties of Protein Tertiary Structure	Proteins	9	1000
Yacht Hydrodynamics	Yacht	7	308

6.2 Regression

In order to illustrate the robust regression IRF model, it is compared with the original RF by using datasets from the UCI Machine Learning Repository [22]. Table 12 shows the number of features m for the corresponding data set and the number of examples N. For every experiment, the same rules as for the numerical classification experiments are used, for example, $N_{\text{tr}} = 10,\ 20,\ 30,\ 40$ training examples are randomly selected from every dataset. The mean square prediction error measure (RMSE) used in experiments is computed as

$$RMSE = \sqrt{N_{\text{test}}^{-1} \sum_{i=1}^{N_{\text{test}}} (y_i - f(\mathbf{x}_i))^2}.$$

To evaluate the RMSE, a cross-validation with 100 repetitions is performed, where $N_{\text{tr}} = 10,\ 20,\ 30,\ 40$ training examples are randomly selected in each run. The quadratic optimization problem is solved using the parameter ε of the set $\mathcal{W}(\varepsilon)$ selected to get the best accuracy. Different values for the regularization hyper-parameter λ have been tested, choosing those leading to the best results. The RF consists of 100 decision trees.

First, the regression models by using the confidence interval (18) for $\alpha = 0.1$ and $\alpha = 0.01$ and the Kolmogorov-Smirnov bounds for the expectation (KS-interval in Tables 13-16) for $\gamma = 0.1$ and $\gamma = 0.01$ are studied. Let F denote the cumulative distribution function associated with the unknown probability measure P of some univariate predicted data $z_1^{(t)}, \ldots, z_N^{(t)}$ on leaves of the t-th tree, and let $\hat{F}_N$ be the associated empirical cumulative distribution function. The Kolmogorov–Smirnov bounds $\underline{F}_N$ and $\overline{F}_N$ for $\hat{F}_N$ are defined as follows [17]:

$$\underline{F}_N(z) = \max(F_N(z) - d_{N,1-\gamma}), 0),$$
$$\overline{F}_N(z) = \min(F_N(z) + d_{N,1-\gamma}, 1).$$

Here $d_{N,1-\gamma}$ is a quantile of the test distribution with the test level $\gamma \in (0, 1)$. The ways for computing $d_{N,1-\gamma}$ for given N and γ can be found in the book [17]. In order to compute bounds, lower and upper expectations corresponding to the functions $\underline{F}_N$ and $\overline{F}_N$, respectively, $a = z_1^{(t)}$ and $b = z_N^{(t)}$ are introduced such that $\underline{F}_N(z) = 1$ for all $z \geq b$ and $\overline{F}_N(z) = 0$ for all $z \leq a$.

Table 13: Comparison of the RF with the IRF by $\alpha = 0.1$, $\alpha = 0.01$ and $\gamma = 0.1$, $\gamma = 0.01$ when 10 examples are used for training

		Confidence interval		KS-interval	
Data set	RF	IRF, $\alpha = 0.1$	IRF, $\alpha = 0.01$	IRF, $\gamma = 0.1$	IRF, $\gamma = 0.01$
Airfoil	6.89	**6.65**	6.67	6.68	6.70
Concrete	15.61	14.96	**14.91**	15.41	15.43
Energy_c	7.09	7.22	7.11	**6.98**	6.99
Energy_h	7.62	7.48	**7.34**	7.74	7.77
Estate	11.13	10.89	**10.68**	10.86	10.90
Proteins	6.34	**6.08**	6.09	**6.08**	6.09
Yacht	11.90	12.04	11.73	**11.66**	11.67

Table 14: Comparison of the RF with the IRF by $\alpha = 0.1$, $\alpha = 0.01$ and $\gamma = 0.1$, $\gamma = 0.01$ when 20 examples are used for training

		Confidence interval		KS-interval	
Data set	RF	IRF, $\alpha = 0.1$	IRF, $\alpha = 0.01$	IRF, $\gamma = 0.1$	IRF, $\gamma = 0.01$
Airfoil	6.26	6.07	6.08	**6.02**	**6.02**
Concrete	13.52	**12.87**	12.89	13.46	13.46
Energy_c	4.34	4.29	4.28	**4.25**	**4.25**
Energy_h	4.66	4.46	**4.40**	4.66	4.66
Estate	9.46	9.17	**9.08**	9.17	9.18
Proteins	6.02	5.86	5.88	**5.78**	5.79
Yacht	7.10	7.19	**6.83**	7.02	7.03

The numerical results for cases with 10, 20, 30, 40 training examples are shown in Tables 13-16, respectively. The best performance for each dataset is shown in bold. It can be seen from Tables 13-16 that the IRF outperforms the original RF for all datasets.

In order to formally show when the proposed IRF outperforms the original RF, the t-test is again applied. The largest values of the IRF accuracy measures for $\alpha = 0.1$ and $\alpha = 0.01$ are compared with the corresponding RF accuracy measure. The t statistic in this case is distributed according to the Student's t-distribution with $7 - 1$ degrees of freedom. The corresponding p-values for cases with 10, 20, 30, 40 training examples are 0.0042, 0.0036, 0.0013, 0.002, respectively. It can be seen from the results that the proposed model outperforms the original RF for all considered datasets. It is interesting to consider the results corresponding to the use of the confidence intervals and the KS-intervals separately. The p-values for cases with 10, 20, 30, 40 training examples using confidence intervals are 0.013, 0.008, 0.008, 0.005, respectively. The same p-values using the KS-intervals are 0.018, 0.014, 0.017, 0.008.

Fig. 4 illustrates how the accuracy measures depend on the training size for datasets:

Table 15: Comparison of the RF with the IRF by $\alpha = 0.1$, $\alpha = 0.01$ and $\gamma = 0.1$, $\gamma = 0.01$ when 30 examples are used for training

		Confidence interval		KS-interval	
Data set	RF	IRF, $\alpha = 0.1$	IRF, $\alpha = 0.01$	IRF, $\gamma = 0.1$	IRF, $\gamma = 0.01$
Airfoil	5.78	5.73	5.77	**5.61**	**5.61**
Concrete	12.21	**11.79**	11.80	12.01	12.00
Energy_c	3.75	3.66	**3.63**	3.69	3.69
Energy_h	3.78	3.64	**3.60**	3.73	3.73
Estate	8.97	8.60	**8.59**	8.79	8.80
Proteins	5.90	5.81	5.86	**5.70**	5.71
Yacht	5.10	4.85	**4.72**	5.15	5.14

Table 16: Comparison of the RF with the IRF by $\alpha = 0.1$, $\alpha = 0.01$ and $\gamma = 0.1$, $\gamma = 0.01$ when 40 examples are used for training

		Confidence interval		KS-interval	
Data set	RF	IRF, $\alpha = 0.1$	IRF, $\alpha = 0.01$	IRF, $\gamma = 0.1$	IRF, $\gamma = 0.01$
Airfoil	5.48	5.38	**5.37**	5.38	5.38
Concrete	11.16	10.91	**10.88**	11.07	11.05
Energy_c	3.36	**3.24**	**3.24**	3.36	3.36
Energy_h	3.22	**3.15**	**3.15**	3.20	3.21
Estate	8.62	**8.37**	8.39	8.48	8.47
Proteins	5.82	5.70	5.74	5.67	**5.65**
Yacht	4.14	4.17	4.10	**4.04**	4.06

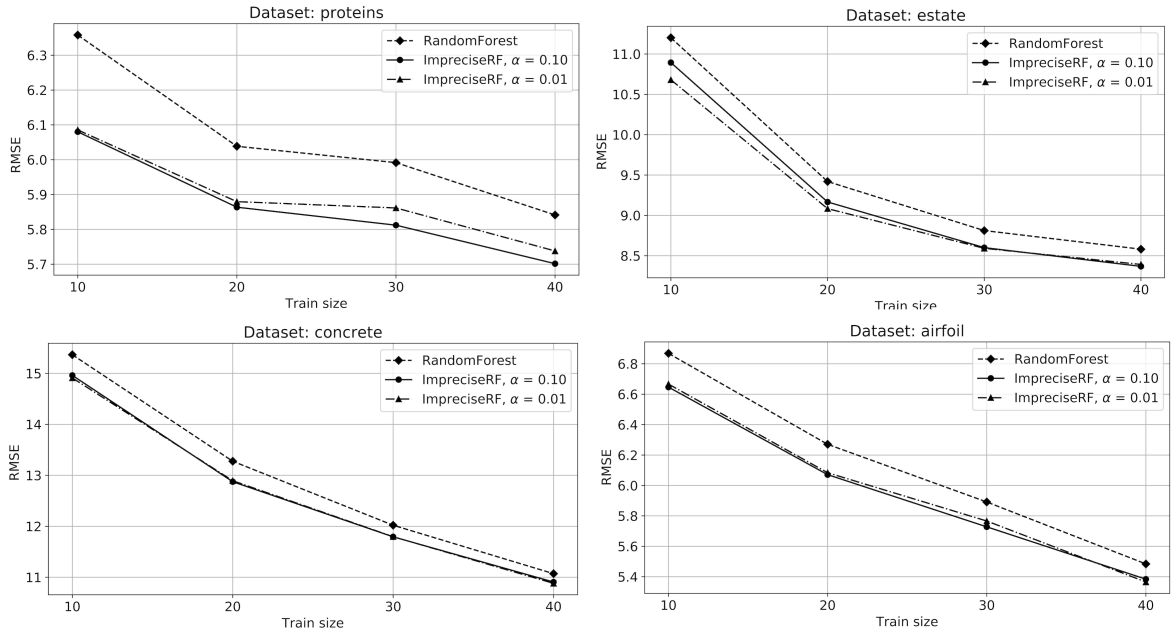

Figure 4: Illustration of the proposed model outperformance for datasets: Proteins, Estate, Concrete, Airfoil

Proteins, Estate, Concrete, Airfoil. The results are taken for cases of the confidence interval usage. The lines with diamond markers, circle markers, and triangle markers correspond to the original RF, the IRF by $\alpha = 0.1$, and the IRF by $\alpha = 0.01$, respectively.

Fig. 6.2 illustrates the same accuracy measures for Proteins and Estate datasets under condition that the testing data are added by the noise.

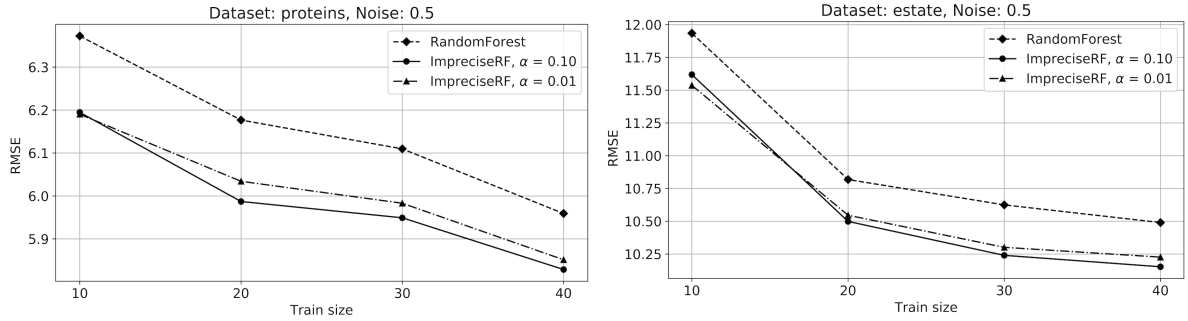

Illustration of the proposed regression model outperformance by adding the noise to testing data for datasets: Proteins, Estate

7 Conclusion

This paper has presented a novel approach for classification and regression using RFs, taking into account a lack of sufficient data to warrant resulting predictions to be precise. It differs from many available approaches to classification and regression because it does not change trees as week learners, but impacts on their weights which are used for combining the tree outputs. Simple linear and quadratic optimization problems have been obtained for computing weights optimal to some extent. Only two types of models have been studied: the classification RF model and the regression RF model. However, the idea underlying the considered models can be extended to other ensemble-based models. This is a direction for further research.

The introduction of new loss functions tried to separate optimization problems in order to simplify the solution and to avoid extremely complex non-linear optimization problems which are unsolvable and not really necessary. Of course, it is necessary to pay for this simplification by the decision being pessimistic, which is logically related to robustness. Perhaps, better results could be obtained by using the original loss functions and solving the corresponding complex non-linear optimization problems. However, our numerical experiments have illustrated that the proposed replacements lead to outperforming results compared to standard RF. It follows from the numerical experiments on the basis of real datasets that the proposed IRF is very efficient when only a few training examples are available and testing examples are noisy. It is also important to note that the proposed model successfully deals with large datasets due to restrictions introduced for weights and due to flexibility of the used imprecise models when intervals of class probabilities or confidence intervals of predicted values strongly depend on the number of training examples.

A disadvantage of the proposed approach is that many additional tuning parameters appear, including parameters of the imprecise models (the contamination parameter, the hyper-parameter of the IDM), the regularization hyper-parameter, and the parameter representing the number of tree groups. As a result, the tuning procedure for selecting the best parameters is hard for training and requires quite substantial additional time. Moreover, a quadratic optimization problem with a large number of variables has to be solved for implementing the meta-model. Its solution also requires additional time.

The proposed approach can be extended to various imprecise statistical models, for example, the imprecise pari-mutuel model, the constant odds-ratio model. In particular, the numerical results have been provided for the Kolmogorov-Smirnov bounds, which can be regarded as a special form of the imprecise pari-mutuel model. These imprecise models have a final set of extreme points of the set of probability distributions and bounds for probabilities of events. This implies that the corresponding minimax optimization problems can be solved in the same way.

It should be noted that only two specific loss functions have been applied to computing the optimal weights. However, functions different from the given ones can be also applied to simplifying the obtained optimization problems and to getting better results. This is a direction for further research. Moreover, it is very interesting to consider distances, which

belong to information-theoretic metrics, for example, the Kullback–Leibler divergence for comparing two class probability distributions.

One of the important and challenging problems of machine learning is the imbalance class problem. It can be considered from two points of view. On the one hand, this problem may occur when examples of one class outnumber the examples of other classes. As a result, one of the classes has a small number of training data. This fact may lead to the problem which has been solved in the paper. On the other hand, the imbalanced data may significantly bias the probabilities of classes, which are used for constructing and for training the meta-model. Therefore, new imprecise statistical model (biased models) should be studied in place of the IDM. The above separate views of the problems as well as their joint consideration are also interesting directions for further research.

Finally, it should be pointed out that the idea of introducing the weights for taking into account imprecision can be extended to different functions than weighted averaging, for example, some non-linear functions of a special form or neural networks which can be also viewed as complex functions of weights.

Acknowledgement

The authors would like to express their appreciation to the anonymous referees whose very valuable comments have improved the paper.

This work is supported by the Russian Science Foundation under grant 18-11-00078.

References

[1] J. Abellan, R.M. Baker, F.P.A. Coolen, R.J. Crossman, and A.R. Masegosa. Classification with decision trees from a nonparametric predictive inference perspective. *Computational Statistics and Data Analysis*, 71:789–802, 2014.

[2] J. Abellan, C.J. Mantas, and J.G. Castellano. A random forest approach using imprecise probabilities. *Knowledge-Based Systems*, 134:72–84, 2017.

[3] J. Abellan, C.J. Mantas, J.G. Castellano, and S. Moral-Garcia. Increasing diversity in random forest learning algorithm via imprecise probabilities. *Expert Systems With Applications*, 97:228–243, 2018.

[4] J. Abellan and S. Moral. Building classification trees using th building classification trees using the total uncertainty criterion. *International Journal of Intelligent Systems*, 18(12):1215–1225, 2003.

[5] M.N. Adnan and M.Z. Islam. Forest cern: a new decision forest building technique. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 304–315, Cham, April 2016. Springer.

[6] E. Alfaro, M. Gamez, and N. Garcia. *Ensemble Classification Methods with Applications in R*. John Wiley & Sons, Incorporated, 2019.

[7] M. Badarna and I. Shimshoni. Selective sampling for trees and forests. *Neurocomputing*, 358:93–108, 2019.

[8] J.-M. Bernard. An introduction to the imprecise Dirichlet model for multinomial data. *International Journal of Approximate Reasoning*, 39(2-5):123–150, 2005.

[9] P. Bonissone, J.M. Cadenas, M.C. Garrido, and R.A. Diaz-Valladares. A fuzzy random forest. *International Journal of Approximate Reasoning*, 51:729–747, 2010.

[10] L. Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[11] M.E.H. Daho, N. Settouti, M.E.A. Lazouni, and M.E.A. Chikh. Weighted vote for trees aggregation in random forest. In *2014 International Conference on Multimedia Computing and Systems (ICMCS)*, pages 438–443. IEEE, April 2014.

[12] J. Demsar. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7:1–30, 2006.

[13] S. Destercke and V. Antoine. Combining imprecise probability masses with maximal coherent subsets: Application to ensemble classification. In *Synergies of Soft Computing and Statistics for Intelligent Data Analysis*, pages 27–35. Springer, Berlin, Heidelberg, 2013.

[14] K. Fawagreh, M.M. Gaber, and E. Elyan. Random forests: from early developments to recent advancements. *Systems Science & Control Engineering*, 2(1):602–609, 2014.

[15] W. Feng, G. Dauphin, W. Huang, Y. Quan, and W. Liao. New margin-based subsampling iterative technique in modified random forests for classification. *Knowledge-Based Systems*, 182(Article 104845):1–12, 2019.

[16] A.J. Ferreira and M.A.T. Figueiredo. Boosting algorithms: A review of methods, theory, and applications. In C. Zhang and Y. Ma, editors, *Ensemble Machine Learning: Methods and Applications*, pages 35–85. Springer, New York, 2012.

[17] N.L. Johnson and F. Leone. *Statistics and experimental design in engineering and the physical sciences*, volume 1. Wiley, New York, 1964.

[18] A. Jurek, Y. Bi, S. Wu, and C. Nugent. A survey of commonly used ensemble-based classification techniques. *The Knowledge Engineering Review*, 29(5):551–581, 2014.

[19] H. Kim, H. Kim, H. Moon, and H. Ahn. A weight-adjusted voting algorithm for ensemble of classifiers. *Journal of the Korean Statistical Society*, 40(4):437–449, 2011.

[20] L.I. Kuncheva. *Combining Pattern Classifiers: Methods and Algorithms*. Wiley-Interscience, New Jersey, 2004.

[21] H. B. Li, W. Wang, H. W. Ding, and J. Dong. Trees weighting random forest method for classifying high-dimensional noisy data. In *2010 IEEE 7th International Conference on E-Business Engineering*, pages 160–163. IEEE, Nov 2010.

[22] M. Lichman. UCI machine learning repository, 2013.

[23] F.T. Liu, K.M. Ting, and Z.-H. Zhou. Isolation forest. In *Eighth IEEE International Conference on Data Mining, ICDM'08*, pages 413–422. IEEE, 2008.

[24] B. Lodhi and J. Kang. Multipath-densenet: A supervised ensemble architecture of densely connected convolutional networks. *Information Sciences*, 482:63–72, 2019.

[25] C.J. Mantas and J. Abellan. Analysis and extension of decision trees based on imprecise probabilities: Application on noisy data. *Expert Systems with Applications*, 41(5):2514–2525, 2014.

[26] P.-A. Matt. Uses and computation of imprecise probabilities from statistical data and expert arguments. *International Journal of Approximate Reasoning*, 81:63–86, 2017.

[27] S. Moral. Learning with imprecise probabilities as model selection and averaging. *International Journal of Approximate Reasoning*, 109:111–124, 2019.

[28] S. Moral-Garcia, C.J. Mantas, J.G. Castellano, M.D. Benitez, and J. Abellan. Bagging of credal decision trees for imprecise classification. *Expert Systems with Applications*, 141(Article 112944):1–9, 2020.

[29] R. Polikar. Ensemble learning. In C. Zhang and Y. Ma, editors, *Ensemble Machine Learning: Methods and Applications*, pages 1–34. Springer, New York, 2012.

[30] Y. Ren, L. Zhang, and P. N. Suganthan. Ensemble classification and regression-recent developments, applications and future directions [review article]. *IEEE Computational Intelligence Magazine*, 11(1):41–53, 2016.

[31] C.P. Robert. *The Bayesian Choice*. Springer, New York, 1994.

[32] J.J. Rodriguez, L. Kuncheva, and C.J. Alonso. Rotation forest: A new classifier ensemble method. *EEE Transactions on Pattern Analysis and Machine Intelligence*, 28(10):1619–1630, 2006.

[33] L. Rokach. Ensemble-based classifiers. *Artificial Intelligence Review*, 33(1-2):1–39, 2010.

[34] L. Rokach. Decision forest: Twenty years of research. *Information Fusion*, 27:111–125, 2016.

[35] L. Rokach. *Ensemble Learning: Pattern Classification Using Ensemble Methods*, volume 85. World Scientific, 2019.

[36] C.A. Ronao and S.-B. Cho. Random forests with weighted voting for anomalous query access detection in relational databases. In *Artificial Intelligence and Soft Computing. ICAISC 2015*, volume 9120 of *Lecture Notes in Computer Science*, pages 36–48, Cham, 2015. Springer.

[37] L.V. Utkin, M.S. Kovalev, and A.A. Meldo. A deep forest classifier with weights of class probability distribution subsets. *Knowledge-Based Systems*, 173:15–27, 2019.

[38] L.V. Utkin and M.A. Ryabinin. A Siamese deep forest. *Knowledge-Based Systems*, 139:13–22, 2018.

[39] P. Walley. *Statistical Reasoning with Imprecise Probabilities*. Chapman and Hall, London, 1991.

[40] P. Walley. Inferences from multinomial data: Learning about a bag of marbles. *Journal of the Royal Statistical Society, Series B*, 58:3–57, 1996. with discussion.

[41] Y. Wang, Y. Li, W. Pu, K. Wen, Y.Y. Shugart, M. Xiong, and L. Jin. Random bits forest: a strong classifier/regressor for big data. *Scientific Reports*, 6(30086):1–7, 2016.

[42] M. Wozniak, M. Grana, and E. Corchado. A survey of multiple classifier systems as hybrid systems. *Information Fusion*, pages 3–17, 2014.

[43] H. Xu, C. Caramanis, and S. Mannor. Robustness and regularization of support vector machines. *The Journal of Machine Learning Research*, 10(7):1485–1510, 2009.

[44] P. Yang, E.H. Yang, B.B. Zhou, and A.Y. Zomaya. A review of ensemble methods in bioinformatics. *Current Bioinformatics*, 5(4):296–308, 2010.

[45] Z.-H. Zhou. *Ensemble Methods: Foundations and Algorithms*. CRC Press, Boca Raton, 2012.